X3T9.2/90-189

A Serial Link for Storage Sub-Systems

I D Judd

IBM UK Laboratories Ltd
Mail Point 200
Hursley Park
Winchester
England SO21 2JN
Tel: 011-44-703-701421
Fax: 011-44-962-842327

4 December 1990

- Development activities in UK Storage Products:

    Hard disks for mid-range systems

    Power and packaging solutions

    Storage sub-systems

    We have experience of IPI-3 and SCSI

- The serial link is a technology effort encompassing:

    Overall sub-system architecture

    Development of a working prototype in CMOS LSI

- The serial link has significant advantages in:

    Sub-system performance

    Packaging, especially for small form-factor devices

    Power dissipation

    Reliability, Availability and Serviceability (RAS)

    Overall sub-system cost

- IBM wants an industry-standard architecture

    Customer requirement for alternate sources

    Wider choice of devices

- A general-purpose link for I/O devices

  Dual-simplex protocol (8 Mbytes/s each way)

  Packet multiplexing allows concurrent I/O operations

  Can support high-level and low-level command sets

  Currently allows point-to-point communication only

  Specified distance is 10 Metres using 2 twisted pairs

- Compact and economical

  Low foot-print for small form-factor devices

  Prototype is fully integrated in 1 micron CMOS

  Low-cost cables and connectors

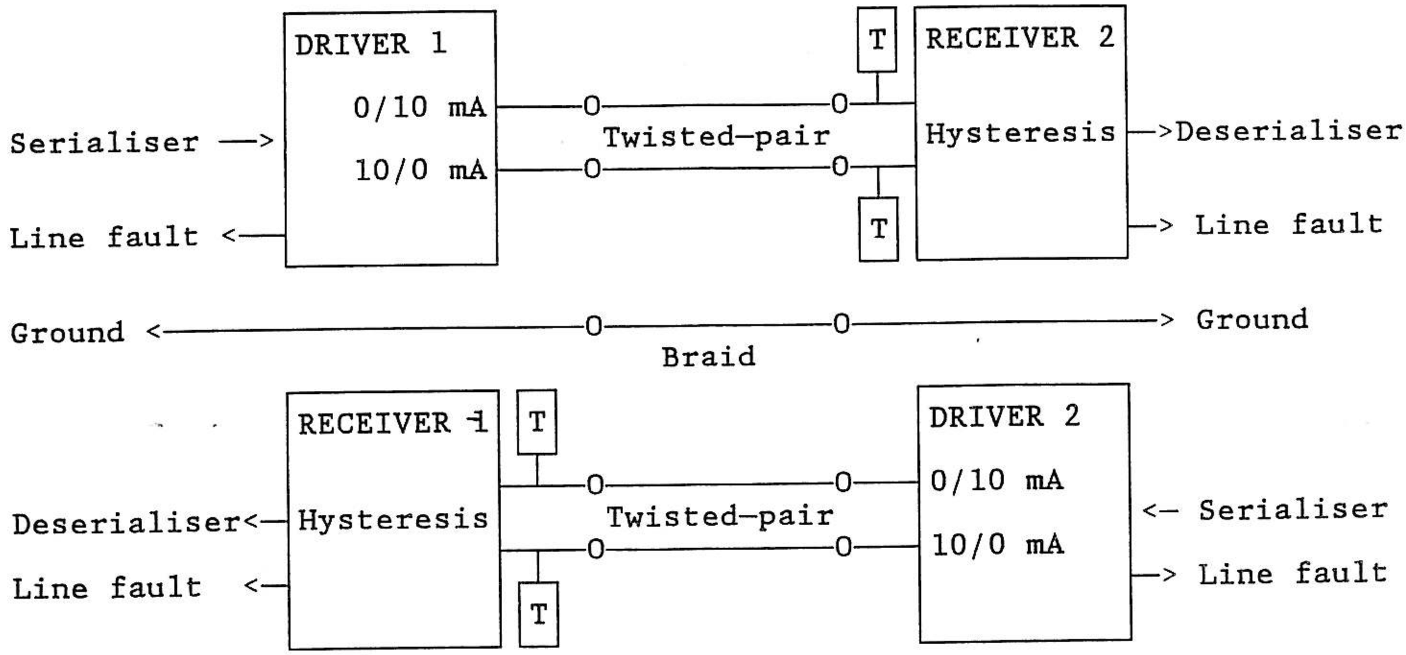
- Excellent Reliability, Availability & Serviceability

  High degree of error detection

  Transparent packet recovery

  Simple fault isolation and concurrent maintenance

  Wrap mode for power-on self-test



- Cable
  - 2 x 100-ohm twisted pairs, 24 AWG
  - Individual foil shields and overall braid
  - 6 mm diameter, 10 Metres maximum length
  - Vendor: Brand-Rex
- Connectors
  - 6-pin latching cable plug and fixed receptacle
  - Vendor: DuPont ('Latch N Lok' series)
- Line driver
  - Differential open-drain current sink
  - +/- 2 Volt common-mode range
- Line receiver
  - Differential comparator with hysteresis
  - Terminators are 51R to +5 Volts

- 10-bit frames at 80 Mbits/s

  Synchronous clocking with NRZI signalling

  Digital clock recovery in deserialiser

  Data frames contain two 5-bit symbols using 4/5 code

SYMBOL	DATA	SYMBOL	DATA
1 1 1 1 0	x'0'	1 0 0 1 0	x'8'
0 1 0 0 1	x'1'	1 0 0 1 1	x'9'
1 0 1 0 0	x'2'	1 0 1 1 0	x'A'
1 0 1 0 1	x'3'	1 0 1 1 1	x'B'
0 1 0 1 0	x'4'	1 1 0 1 0	x'C'
0 1 0 1 1	x'5'	1 1 0 1 1	x'D'
0 1 1 1 0	x'6'	1 1 1 0 0	x'E'
0 1 1 1 1	x'7'	1 1 1 0 1	x'F'

PROTOCOL FRAMES	MEANING	
1 0 0 0 1 0 0 1 0 0	FLAG	(Delimiter & sync)
0 1 1 0 1 0 1 1 0 1	ACK	(Acknowledgement)
1 1 1 1 1 1 1 1 1 1	RR	(Pacing)
1 1 0 0 1 1 1 0 0 1	NUL	(Pad)

Packet format

FLAG	CONTROL	ADDRESS	DATA .... DATA	CRC	CRC	FLAG

- FLAG
  - A protocol frame delimiting the start and end of a packet
  - Trailing FLAG can also be the leading flag of next packet
  - FLAG does not occur elsewhere in any bit phase
  - Sent continuously when the line is idle
- CONTROL FIELD
  - A single data frame managed by the transport layer:
    - 2-bit Packet Sequence Number
    - Link Reset bit for error recovery
    - Total Reset bit
    - 4 user-definable bits
- ADDRESS FIELD
  - A single data frame specifying the packet source/destination
  - Supplied by the application and used by the transport layer
- DATA FIELD
  - From zero up to some maximum number of data frames
  - Of interest only to the application
- CRC FIELD
  - 2 data frames to check all frames between the FLAG's
  - Managed by the transport layer

Packet types

- Control packets

FLAG	CONTROL	ADDRESS	CRC	CRC	FLAG

Used for hardware resets

Identified by bits in control field

- Message packets

FLAG	CONTROL	ADDRESS	MESSAGE	CRC	CRC	FLAG

Used for commands, status and initiating data transfers

Identified by pre-assigned destination address(es)

Interrupt the microprocessor in the destination node

- Data packets

FLAG	CONTROL	ADDRESS	DATA	CRC	CRC	FLAG

Used for data transfer

Address is often allocated dynamically using messages

Source & destination are often hardware DMA channels

- Packets are transferred in 'dual simplex' mode
  - A node can transmit and receive packets simultaneously
  - Inbound and outbound packets are treated independently
  - Both nodes are peers, ie. the **link** is symmetric
- A node acknowledges valid inbound packets:
  - Source node sends the trailing FLAG to finish a packet
  - Destination node must send an ACK within 10 us
  - Source node can then reuse the outbound packet buffer
- A node paces each inbound packet:
  - Source node sends the control field of a packet
  - Destination sends an RR only when ready for next packet
  - Source node can then send another packet
- ACK and RR are protocol frames, not packets
  - Duplicated for checking
  - Can be interleaved within a packet to reduce latency
- Source can **start** the next packet while waiting for ACK
  - Must not send the trailing FLAG if still waiting for ACK
  - Send NUL frames instead until ACK received or time-out
  - Provides an unambiguous relation between ACK and packet

- To optimise cost/performance the implementation defines:

  Maximum packet size

  Number of packet buffers

-

PACKET SIZE	SIMPLEX	DUAL-SIMPLEX
16 bytes	6.1 MB/s	2 x 5.1 MB/s
32 bytes	6.9 MB/s	2 x 6.2 MB/s
64 bytes	7.4 MB/s	2 x 7.0 MB/s
128 bytes	7.7 MB/s	2 x 7.5 MB/s

- Minimum for slave node (Master polls for asyncs.):

  1 floating buffer for transmit or receive

- High-speed slave node (Master polls for asyncs.):

  A/B floating buffers for transmit or receive

- High-speed dual-simplex node:

  A/B buffers dedicated for transmit

  A/B buffers dedicated for receive

- Dedicated message buffers are also useful if:

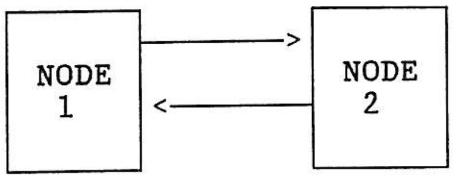
  A node has no other buffering, and,

  Real-time data transfers are essential

  eg. for hard disks with a low-level command set

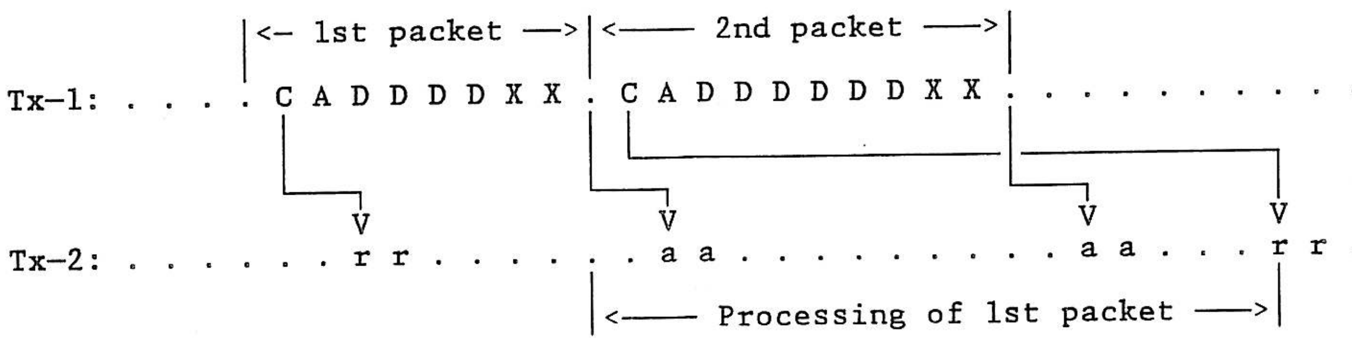
Protocol examples

- Nomenclature

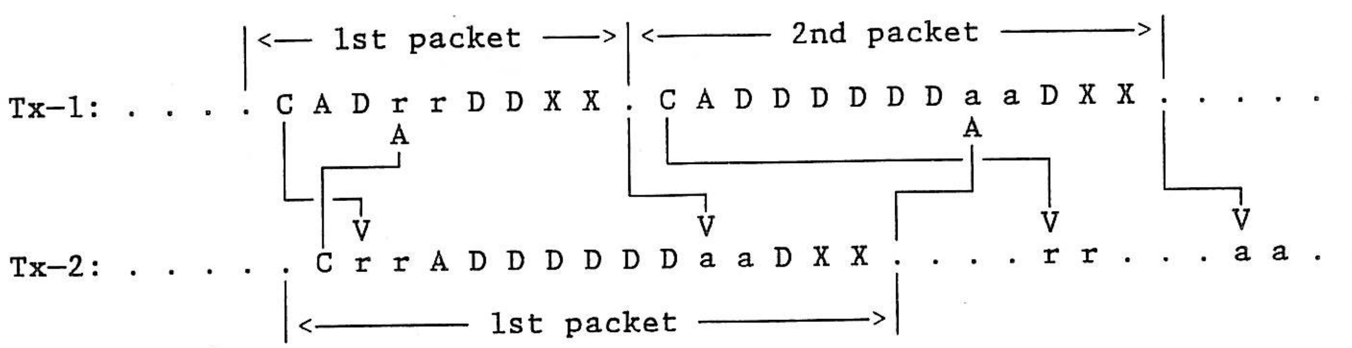


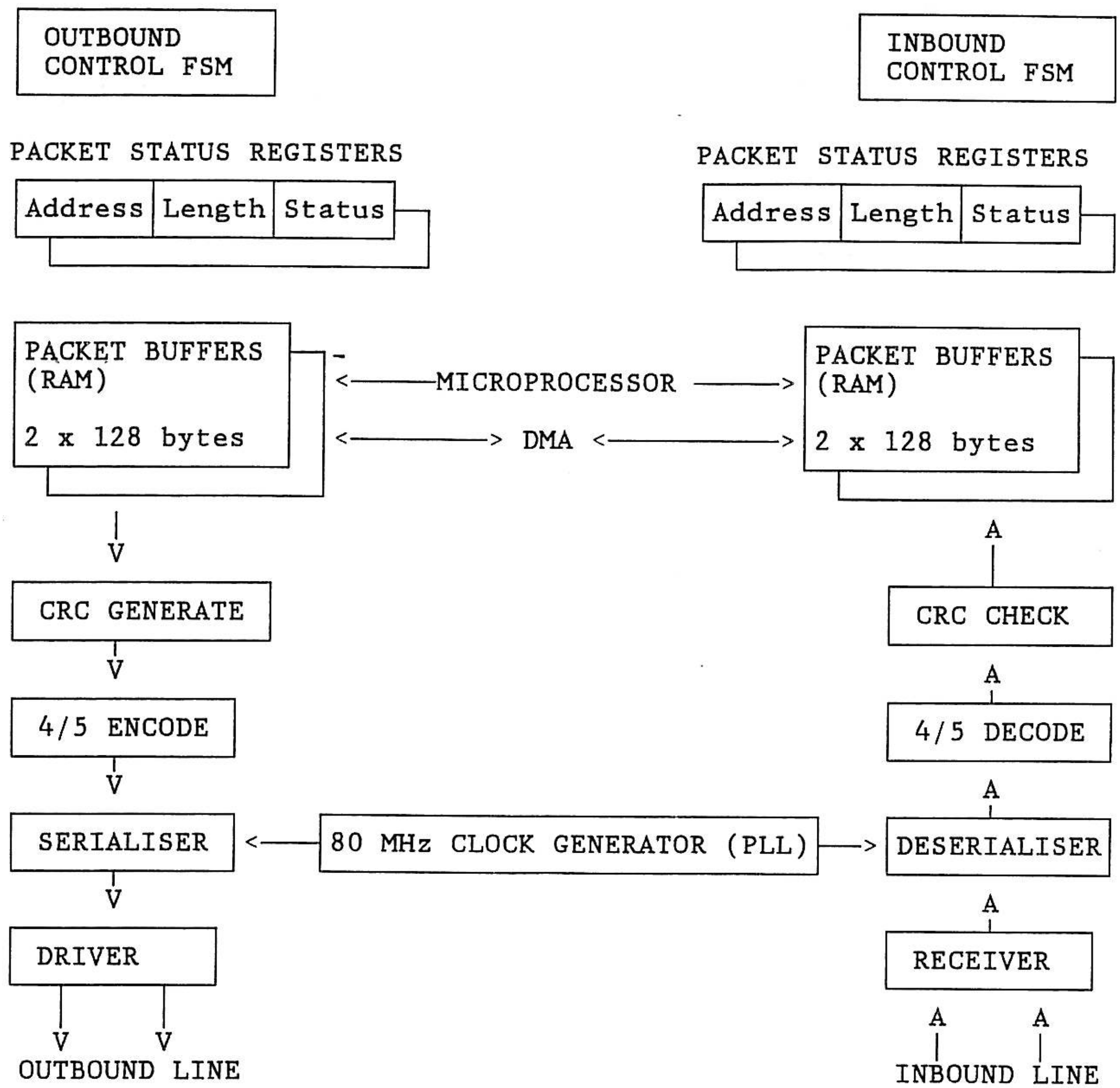
C – Control
A – Address
D – Data
X – CRC
. – FLAG
a – ACK
r – RR

- Simplex transfer with A/B buffering



- Dual-simplex transfer with A/B buffering





- Approximately 10K equivalent gates
- Line driver/receiver and PLL are analogue circuits

- 'Wrap' provides an excellent power-on self-test
    - Serialiser output is switched to local deserialiser input
    - Can also exchange 'wrap' messages with remote node
- The hardware provides comprehensive error detection
    - Line faults
    - Illegal frames
    - CRC errors
    - Non-sequential packet sequence numbers
    - Protocol errors
    - ACK time-outs
- When a node detects an error:
    - Transmission stops at the end of the current packet
    - The hardware enters the 'check' state
    - An Error Recovery Procedure (ERP) is invoked
- The link ERP is architected
    - Avoids potential incompatibilities
    - Transparent to the application (If successful)
    - Minimises the impact of errors if the link is shared

- Each node recovers errors on its own outbound line
- Each node maintains a Transmit Sequence Number (TSN)

  2 bits, incremented modulo 4 for each packet sent

  Included in each packet as Packet Sequence Number (PSN)
- Each node maintains a Receive Sequence Number (RSN)

  2 bits, incremented modulo 4 for each ACK sent

  Compared with PSN in received packets
- A 'Link Reset' control packet is defined

  Contains the RSN in the address field

  Forces the destination node into the 'check' state
- When a node enters the 'check' state it invokes the ERP:

  Transmit a Link Reset (Contains local RSN)

  Wait to receive a Link Reset (Contains remote RSN)

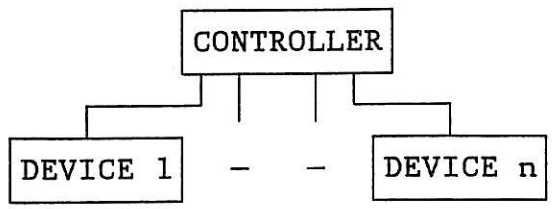
  Compare local TSN and remote RSN

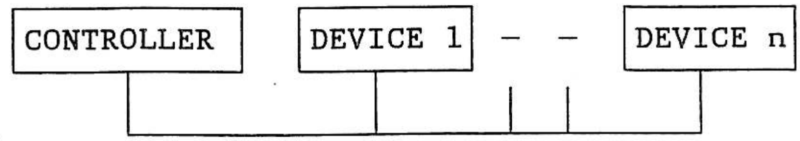
  Discard outbound packets corresponding to any lost ACK's

  Restore hardware to 'ready' state

  Retransmit any lost packets from outbound buffers

- We chose point-to-point in preference to multi-drop:





- Point-to-point
  - + Overall simplicity
  - + Better RAS characteristics:
    - Fault isolation
    - Inherent concurrent maintenance
  - - More ports altogether
    - 2N rather than N+1
    - May also need dual-ported controllers/devices
  - - Cable congestion at controller
- Multi-drop
  - + Can readily attach more devices
  - + Inherent peer-to-peer communication
  - - Needs higher bandwidth; may be difficult due to:
    - Technology break-points
    - Increased cost of each node
    - Transmission line is degraded by stubs
    - Cable must be longer overall
  - - Extra overheads:
    - Arbitration to resolve contention
    - Resynchronisation of deserialisers
    - Queuing for a single interface
  - - Need data buffer in device to avoid lost revolutions
  - - Conflicts with read-ahead to buffer in controller
  - - Not transferable to an optical medium

Higher data rates

- Future disk applications will need higher speed due to

  Technology advances (BPI, RPM, parallel heads)

  Disk arrays (Striping)

  Smaller form-factors (Higher controller fan-out)

- Limiting factors for a faster link:

  Logic delays, especially in the deserialiser

  Rise/fall times of line driver and receiver

  High-frequency losses in the cable

- 8 MB/s is proven today using,

  1 micron CMOS (1 ns loaded gate delay)

  Twisted pair cabling up to 20 Metres

- Up to 20 MB/s should be feasible in 1991 using:

  0.7 micron CMOS (0.4 ns loaded gate delay)

  Improved/shorter cable, OR,

  Low-cost fibre optics

- The interface must not limit performance

  Low-level orders to reduce device over-head

  Device sends raw read data to avoid latency

  No data buffer in the device

  Read-ahead to controller buffer & terminate quickly

  Zero-latency reads and writes

  Back-to-back writes

- Read and write orders can access multiple sectors

  No critical paths in the array controller

  Gaps are not constrained by link turn-around

- No device-dependent **hardware** above the interface

  eg. ECC check/generate should be in the device

  Common controller hardware for a range of devices

- Integrated spindle synchronisation

  Controller broadcasts a special control packet

  Controller has Rotational Position Knowledge (RPK)